Concurrent Real-time Schedulers, a Classification Based on Functions

Pedro Guevara-López[1], Oscar A. Morales-Moreno[2], José S. Falcón-López[3]

[1] National Polytechnic Institute - Higher School of Mechanical and Electrical Engineering, Santa Ana 1000, Mexico City, C. P. 04430, Mexico, pguevara@ipn.mx

[2] National Polytechnic Institute - Research Center for Applied Science and Advanced Technology, Legaria 964, Mexico City, C. P. 11500, Mexico, oscarabimael@hotmail.com

[3] Cuautitlán Izcalli Institute of Technology, Nopaltepec S/N, Estado de Mexico, C. P. 54748, Mexico, jsfalcon68@hotmail.com

Abstract: Real-time Systems (RTS) are generally implemented into digital computers with Real-time Operating Systems (RTOS) where all activities are performed by a set of Concurrent Real-time Tasks (CRTT) which requires attention by a processor through a Concurrent Real-time Task Scheduler (CRTTS). In this sense, this concept is commonly considered as a resource allocator to tasks by means of preset algorithm. However, a RTTCS is a function that maps the Set of Arrival Times (ATS) to the Starting Times Set (STS) of the instances related to CRTTS; this conception is general and independent from the scheduling algorithm used (RM, EDF, FIFO, etc.) and it will be useful to define the working environment of RTS in the analysis of schedulability, reconstruction, fail-safe and predictability. It is also useful to know the scheduler characteristics and determine the type of scheduler we are using in the System.

Keywords: Real-time Systems, Real-time Tasks, Real-time Schedulers, Arrival Time, Time Constraints.

1. Introduction

In order to be able to implement an RTS in a digital computer, it is required to have a specific Operating System, in this case, a Real-time Operating System (RTOS); and in this operating system, everything is managed through concurrent Real-time Tasks (RTT). These tasks are in charge of performing all the RTS activities, they have temporary restrictions imposed by the real world and they compete for having processor resources at different temporary intervals; and the one that decides what task will take such resource and when, is the Concurrent Real-time Tasks Scheduler (CRTTS). This paper originates from the need of being able to properly schedule by using a scheduler, since it will help for troubleshooting and to improve scheduling through this rating and behavior analysis.

Within this research area, authors agree on a definition: "Concurrent Real-Time Tasks Scheduler (CRTTS) is an algorithm that allocates processor resources to different areas at different moments" remembering that an algorithm is an orderly and finite set of calculations that allow to find a solution to a problem. This concept is informal, not defining resources, inputs and outputs. For this context, in this work we formally define a Concurrent Real-time Tasks Scheduler. Scheduling algorithms mentioned above are very important to be able to make a general rating of schedulers. RM algorithm (Rate Monotonic algorithm) as per [1], is a scheduler with fixed priorities that consists of allocating the highest priority to the tasks having the shorter period in the system, and the lowest priority to the tasks with longer periods. A problem with this algorithm is a use of resource lower than 100%. According to [2], EDF algorithm is a dynamic scheduler that selects tasks according to their absolute terms. Tasks with short terms will be executed with high priorities, and tasks with long terms will have low priorities. An advantage of this algorithm is that the usage of the resource is almost of 100% for all the tasks set. FIFO algorithm as per [3], this type of scheduling is executed on demand; the first task that arrives is the first one to be executed, until the last one is finished.

2. Real-time Systems

The concept of Real-time is not equivalent to fast or instantaneous. The difference between a real-time system and a fast system is that the latter produces an output without considering the time restrictions of the environment that it interacts with, because, for that type of systems, the time in which data arrive is not the important thing, but the time when the output is produced. In contrast, a RTS must comply with

three conditions: interaction with real world, issuing of correct answers and compliance of temporary restrictions, this is, synchrony with the dynamics of real world. Formally, a RTS according to [4] is defined as:

Definition 1 (Real-time System). A Real-time System (RTS) is the system that generally interacts with real world, where the time when output is produced is significant.

In general, input corresponds to some moment τ_1 of the physical world and output corresponds to a moment τ_2; the time differential between input and output of the system is provided by a processing time c; this is: $\tau_1 = \tau_2 + c$, and this processing time must be small enough not to alter negatively the real world, so limiting it to an interval $(0,\delta]$, with $c \in (0,\delta]$ and δ a time period to consider as a punctual response.

When a RTS is installed in a digital computer, it interacts with real world through conditioning (sensors, actuators, Analog/Digital (A/D and Digital/Analog D/A converters) and it processes its requests through concurrent tasks with temporary restrictions. In general, each of the variables from the dynamic process (inputs, outputs and statuses) is related to a specific Real-time Task; if it is a multi-variable system, then there will be a set of Concurrent Real-time Tasks which require to be scheduled.

3. Real-time Tasks

In accordance to [4] a task is a set $\{J_i, i=1,....,n\}$ of basic activities executed in an operating system and that is why, it must comply with some characteristics that according to [4-5] a task is created when a software program is read from the disk and loaded into the memory to execute it. When it happens, three segments are created: code, data and pile; the first one is a copy of the read file, the second one is the space where task variables are saved and the third one is where operations are performed. In this context and according to [4, 6, 7, 8] the following concepts may be seen in Figure 1.

A Real-time Task (RTT) is an executable working entity J_i that is at least characterized by an arrival time and a temporary restriction and it is formed by a set of instances $J_i = \{j_{i,k}\}$; i is the RTT index and p is the number of tasks, k is the instance index and n is the number of instances with i= (1,…, p), k= (1,…, n), $i, k, n, p \in Z^+$. An instance $j_{i,k}$ is a work unit of a task J_i, defined with $j_{i,k} = (l_{i,k}, c_{i,k}, d_{i,k})$, $i,k \in Z^+$ where $l_{i,k}$ is the arrival time of each instance, $c_{i,k}$ is the instance execution time or computing time and

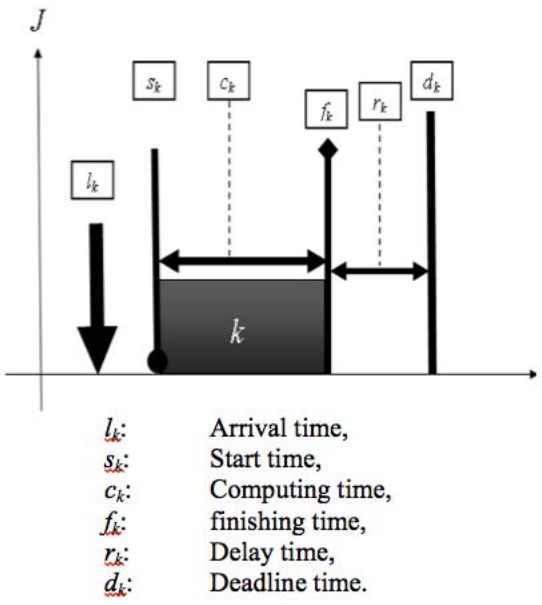

Figure 1. Time constraints of a Real-time Task.

$d_{i,k}$ is the deadline time. The absolute arrival time l_k of an instance $j_{i,k}$ of a RTT J_i is defined as the time in which the instance ask for attention from the processor in relation to the reference source. Starting time s_k of an instance $j_{i,k}$ of a RTT J_i is defined as the moment in which the instance is given attention from the processor in relation to the temporary reference source. Executing time c_k is the time in which instance $j_{i,k}$ of a RTT J_i concludes its operations, not considering its removals at the processor. The maximum absolute deadline $d_{k-\max}$ is the upper temporary limit in relation to the temporary reference source, before which, instance $j_{i,k}$ of a RTT J_i must end. Operating time $o_{i,k}$ is the difference existing between the arrival time $l_{i,k}$ and the start time $s_{i,k}$ of the instance $j_{i,k}$ of RTT J_i; this is $o_{i,k} = s_{i,k} - l_{i.k} \forall i, k, p \in Z^+$. Delay time $r_{i,k}$ is the difference existing between finishing time $f_{i,k}$ and the absolute maximum term $d_{i,k}$ of instance $j_{i,k}$ of a RTT J_i; this is $r_{i,k} = f_{i,k} - d_{i.k}$. All with $i,k,n,p \in Z^+$.

4. The Concept of Real-time Task Scheduling

Scheduler algorithms have several applications like those presented in [9] that say that scheduling has many applications, varying from metallurgy, chemistry, agro-food or pharmaceutical industries and [10] that addressed the problem of scheduling in flowshop manufacturing systems. In this sense, according to [11], *scheduling* of a Real-time System consists of allocating tasks to the processor: the bi-univocal relation between actions and processors is a *schedule plan* and the module performing this is the *scheduler;* for that it uses a *scheduling algorithm.* As per [12], in order to be able to allocate the tasks corresponding to a Real-time System, it is essential to have a scheduler, since it is the one in charge of giving to the tasks the priority needed to be scheduled. According to [1] "*a scheduling algorithm is a set of rules which determine the task to schedule at a certain moment*".

A formal definition of Concurrent Real-time Tasks Scheduler is the one given in book [2]; herein defines the scheduler as a processor allocator of *P* and resources from *R* for tasks *J*, in order to complete all tasks in the framework of imposed restrictions. The same author says the scheduler is like a "function":

$$\sigma = R^+ \rightarrow N$$
$$\forall t \in R^+, \exists t_1, t_2 \text{ such as } t \in [t_1, t_2) \quad (1)$$
$$\text{and } \forall [t_1, t_2) \ \exists! \sigma(t)$$

In other words $\sigma(t)$ is defined as a step function and $\sigma(t) = k$, with $k<0$, meaning that task J_k is scheduled in *t*, when $\sigma(t) = 0$ means that the CPU is idle.

In order to be able to go to the new concept of scheduler according to [13], you must consider the following definitions to form the new concept, and they are as follows:

Definition 2 (Concurrent Real-time Tasks Set). A Concurrent Real-time Tasks Set (CRTTS) **J** is the set formed by at least *p* Real-time Tasks which compete for *q* resources or processors with the restriction that $p>q$ with $p, q \in Z^+$.

Definition 3 (Arrival Times Set). It is a set **J** of *p* Concurrent Real-time Tasks, each task having *n* instances; $\mathbf{\Pi_k}$ is the Set of arrival times (CTA) in index *k* of $j_{i,k}$ instances.

Definition 4 (Start Times Set). It is a set **J** of *p* Concurrent Real-time Tasks, each task having *n* instances; $\mathbf{S_k}$ is the set of Start Times (CTI) in index *k* of $j_{i,k}$ instances.

The new concept of Concurrent Real-time Tasks scheduler (CRTTS) as per [13-14], is defined with a function $\boldsymbol{\sigma}(k)$ which maps the Set of Arrival Times $\mathbf{\Pi_k}$ to the Set of Start Times $\mathbf{S_k}$.

$$\boldsymbol{\sigma}(k): \mathbf{L_k} \rightarrow \mathbf{S_k}, \text{ con } k \in \boldsymbol{Z}^+. \quad (2)$$

In accordance to the previous definition, any Concurrent Real-time Tasks Scheduler, either Rate Monotonic (RM), Earliest Deadline First (EDF), First Input First Output (FIFO), etc. must comply with the Definition, and it shall be a function mapping of ATS $\mathbf{\Pi_k}$ to STS $\mathbf{S_k}$. Figure 2 is a representative scheme.

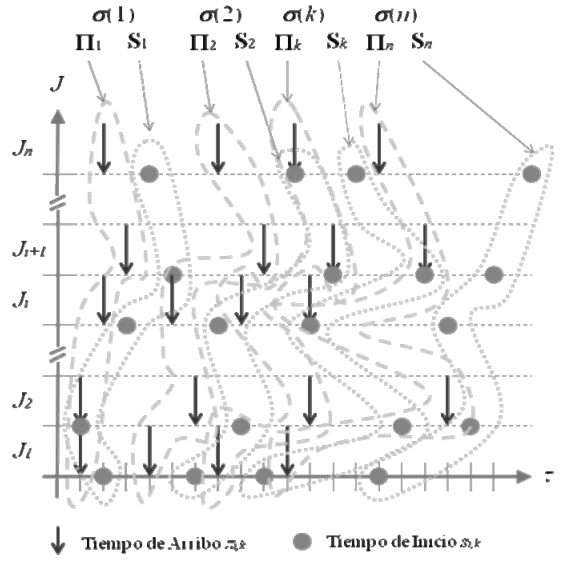

Figure 2. Scheme of the Arrival Times Set Π_k, Start Times Set S_k and Concurrent Real-time Tasks Scheduler $\sigma(k)$.

5. Concurrent Real-time Schedulers, a Classification Based on Functions

Scheduling algorithms, besides complying with the definition must look forward to complying with restriction $f_{i.k} \leq d_{i.k} \forall f_{i.k} \in \mathbf{F_k} \wedge d_{i.k} \in \mathbf{D_k}$. In other words, it is worth saying that it is not enough to map the Set of Arrival times to the Set of Start Times, but it is also necessary to search, for every case, for the compliance of each instance $j_{i,k}$ in all the Set of Concurrent Tasks **J**; however, this task is difficult, considering that there are many scheduling algorithms and that it is a **NP** complete problem.

It may also be seen that the scheduler $\sigma(k)$ is in function of index k; which means that it is not limited to a single scheduling algorithm, the algorithm may vary for every mapping from $\mathbf{L_k}$ to $\mathbf{S_k}$.

Types of Real-time Task Schedulers

As a result of the definition stated in [13] the following definitions were obtained for Real-time Tasks Schedulers.

Definition 5 (Critical Concurrent Real-time Tasks Scheduler). A Critical Concurrent Real-time Tasks Scheduler (CCRTTS) is a function $\sigma(k)$ that maps from the Arrival Times Set $\mathbf{L_k}$ to Start Times Set $\mathbf{S_k}$ such as $\#(\mathbf{L_k}) = \#(\mathbf{S_k}) \wedge \#(\mathbf{F_k}) = \#(\mathbf{D_k}) \; \forall i, k \in \mathbf{Z^+}$.

Definition 6 (Non- Critical Concurrent Real-time Task Scheduler). A Non- Critical Concurrent Real-time Tasks Scheduler (NCCRTTS) is a function $\sigma(k)$ which maps from the Arrival Times Set $\mathbf{L_k}$ to the Start Times Set $\mathbf{S_k}$ such as $\#(\mathbf{L_k}) \geq \#(\mathbf{S_k}) \vee \exists f_{i,k} > d_{i,k} \forall i, k \in \mathbf{Z^+}$.

Definition 7 (Static Concurrent Real-time Task Scheduler). A Static Concurrent Real-time Task Scheduler (SCRTTS) is a function $\sigma(k)$ which maps from the Arrival Times Set $\mathbf{L_k}$ to the Start Times Set $\mathbf{S_k}$ and that never modifies its scheduling algorithm no matter the changing dynamics of ATS $\mathbf{L_k}$.

Definition 8 (Adaptive Concurrent Real-time Task Scheduler). An Adaptive Concurrent Real-time Task Scheduler (ACRTTS) is a function $\sigma(k)$ which maps from the Arrival Times Set $\mathbf{L_k}$ to the Start Times Set $\mathbf{S_k}$, being able to modify its scheduling algorithm and adapting it to the changing dynamics of ATS $\mathbf{L_k}$.

Definition 9 (Optimal Concurrent Real-time Task Scheduler). Optimal Concurrent Real-time Task Scheduler (OCRTTS) is a unique function $\sigma(k)$ which maps from the Arrival Times Set $\mathbf{L_k}$ to the Start Times Set $\mathbf{S_k}$, such that cardinality of both sets is equal, there is a minimum operating time $o_{i.k}$ and there is a maximum delay time $r_{i.k}$; this is: $\exists!\ \sigma(k)$ such that $\#(\mathbf{L_k}) = \#(\mathbf{S_k}) \wedge \exists \min(o_{i.k}) \wedge \exists \max(r_{i.k}) \; \forall i, k \in Z^+$; meaning that it is the only scheduler capable of sending the scheduling of all instances of the Concurrent Real-time Tasks Set **J,** by minimizing the difference between Start Times $l_{i,k}$ and arrival times $s_{i,k}$ and maximizing the difference between maximum terms $d_{i,k}$ and ending times $f_{i,k}$ during all the system evolution.

Definition 10 (Predictive Concurrent Real-time Task Scheduler). A Predictive Concurrent Real-time Task Scheduler (PCRTTS) is a function $\sigma(k)$ which maps from the Arrival Times Set $\mathbf{L_{k+r}}$ to the Start Times Set $\mathbf{S_{k+r}}$, this is: $\sigma(k)$: $\mathbf{L_{k+r}} \rightarrow \mathbf{S_{k+r}} \forall k$; this is, that index k, $\sigma(k)$ is capable of scheduling for index $k+r \leq n$, with $r \in Z^+$.

Definition 11 (Preemptive Concurrent Real-time Task Scheduler). A Preemptive Concurrent Real-time Task Scheduler is a scheduler where at least one instance $j_{i,k}$ of a Real-time Task J_i suspends its scheduled time

$c_{i,k}$ due to the start of an instance $j_{x,y}$ of a Real-time Task J_x with higher priority, with $i, k, x, y \in Z^+$

Comment 1. Each of the segments of an instance $j_{i,k}$ is called inter-instances $j_{i,k,h}$; this is $j_{i,k}=\{ j_{i,k,h}\}$ with $h=1\ldots v$ *e* $i, k, h, v \in Z^+$;. h is the inter-instance index and v is the number of inter-instances of $j_{i,k}$. The value of v will depend on the number of removals that instance $j_{i,k}$.

Comment 2. Computing time $c_{i,k}$ of an instance removed by priorities is the sum of computing times of their inter-instances $c_{i,k}$, this is:

$$c_{i,k} = \sum_{1}^{v} c_{i,k,h} \quad (3)$$

Comment 3. In case of a Preemptive Scheduler, for an Instance $j_{i,k}$ it will only consider as Arrival and Start times those where the instance asks for resources for the first time. This means that the time where attention is demanded by inter-instance $j_{i,k,1}$.

Comment 4. In case of a Preemptive Scheduler, for an Instance $j_{i,k}$ Ending Time is only considered as the one where the instance releases the resources for the last time. This means that it is the time where there is release or ending of inter-instance $j_{i,k,v}$.

6. Conclusions

As a result of this paper, we obtained formal definitions such as: the formal definition of Concurrent Real-time Task Scheduler, as a function which maps from the Arrival Times Set $\mathbf{\Pi}_k$ to the Start Times Set $\mathbf{S}_k$, so that it may be able to have better results and comply with the needs required within a system to be scheduled. Also several new definitions of different types of Schedulers were given, thus being able to determine the type of Scheduler that may apply. According to this generalization, any Concurrent Real-time Task Scheduler, either Rate Monotonic (RM) scheduling algorithm, Earliest Deadline First (EDF), First Input First Output (FIFO) must follow this definition. The actual impact of this rating will be used to determine the behavior of a Scheduler through the analysis of schedulability, reconstruction, fail-safe, and predictability.

Acknowledgements

To National Polytechnic Institute. To Cuautitlán Izcalli Institute of Technology.

REFERENCES

1. LIU, C., J. LAYLAND, **Scheduling Algorithms for Multiprogramming in Hard Real-Time Environment**. Journal of the ACM, Vol. 20, No. 4, 1973, pp. 273-250.
2. BUTTAZZO, G. **Hard Real-Time Computing Systems,** Scuola Superiore S. Anna, Kluwer Academic Publishers, 1997.
3. QNX **System Architecture System Guide**. QNX Software System Ltd, 1997
4. MEDEL, J. J., P. GUEVARA, D. CRUZ, **Temas Selectos de Sistemas en Tiempo Real**. Publicado por la editorial Politécnico Registro de Derechos de Autor: 03-2003-012912240400-0,1 ISBN: 978-970-36-0466-1. México Diciembre de 2007.
5. MÁQUEZ, G. F. **UNIX Programación avanzada**. Universidad de Alcalá de Henares (Madrid), Addison Wesley Iberoamericana, 1993.
6. CRUZ, D., **Modelo Dinámico para Tareas en Tiempo Real**, tesis de Maestría en Ciencias de la Computación, CIC-IPN, 2004. México, Agosto de 2004
7. CRUZ, D. **Modelo para Tiempos de Arribo de Tareas en Tiempo Real Concurrentes**, tesis de Doctorado en Tecnología Avanzada, CICATA-IPN, 2007. México, Diciembre.
8. CRUZ, D., P. GUEVARA, J. MEDEL, **Modelo para Tiempos de Arribo de Tareas en Tiempo Real Concurrentes**. Revista Computación y Sistemas, ISSN 1405-546, Vol. 12 No. 4, México, Abril.
9. BOUKEF, H., M. BENREJEB, P. BORNE, **Flexible Job-shop Scheduling Problems Resolution Inspired from Particle Swarm Optimization**. Studies in Informatics and Control, Volume 17, Issue 3, Bucharest, Romania, 2008, pp. 241-252.
10. RAJKUMAR, R., P. SHAHABUDEEN, P. NAGARAJ, S. ARUNACHALAM, T. Page, **A Bi-Criteria Approach to the M-machine Flowshop Scheduling Problem**.

Studies in Informatics and Control, Volume 18, Issue 2. Bucharest, Romania, 2009, pp. 127-136

11. MEJIA, P., **Curso de Sistemas En Tiempo Real** (Sistemas Operativos). CINVESTAV-IPN, Sección de Computación, México D. F. 2008.
12. GUEVARA, L. P., J. J. MEDEL J., **Introducción a los Sistemas de Tiempo Real.** Editorial Politécnico, registro de Derechos de Autor 03-2003-012912240400-01, México, 2003.
13. MORALES, O., **Formalización del Concepto de Planificador de Tareas en Tiempo Real Concurrentes**, Tesis de Maestría en Tecnología Avanzada, CICATA-IPN.
14. GUEVARA, P., G. DUCHEN, O. MORALES, **El Concepto de Planificador de Tareas en Tiempo Real Basado en Funciones.** CIINDET 2009 VII Congreso Internacional en Innovación y Desarrollo Tecnológico, 7 al 9 de octubre de 2009, Cuernavaca, Morelos. México.
15. GUEVARA, P., O. MORALES, J. FALCON, **Clasificación Basada en Funciones para Planificadores de Tareas en Tiempo Real Concurrentes**, XIV Congreso Latinoamericano de Control Automático y XIX Congreso de la Asociación Chilena de Control Automático, Santiago de Chile, Agosto de 2010.